SPEECH COMPRESSION USING AUTO ENCODERS

Sahaj Singh Maini Siva Charan Mangavalli

Indiana University

ABSTRACT

MP3 and AAC compression have made it easy to compress the high-quality audio to retain audio content perceivable by humans. In this project, we plan to implement Deep Learning based architectures in order to learn the latent representation of the speech signal data to compress the signals and be able to reconstruct the signals. This specific task can be accomplished by using autoencoders since they are ideal choice to learn the latent representations for compression in an unsupervised fashion. This would enable us in recovering the signal with minimal loss in quality. Models that we propose for speech compression are comparatively small models with low inference time. These can be used in systems where interpretability and low latency are given priority.

1. INTRODUCTION

Audio compression has enabled faster adoption of musical services and revolutionized the way audio is stored, transferred and used by parties across the world. Popular audio compression formats like mp3, AAC are designed to retain human perceivable content in the data and discard the rest. There are primarily two dimensions of audio storage files i.e. Compression and loss. Current implementations are a combination of a balance between compression and loss.

- Uncompressed and Lossless formats

  WAV and AIFF come under this format. These formats are an exact copy of the source data without any loss in the quality. The individual files can consume a lot of space. It is ideal for high quality archiving and editing workflows acting like a master copy of the data.

- Compressed and Lossy formats:

  Most common formats like mp3 and AAC fall under this category. A smaller size means faster transfers and the minimum storage requirement. There is some reduction in audio quality owing to selective focus or human-perceivable components.

- Compressed and Lossless formats:

  FLAC and ALAC fall under this category. They have reduced file size without having to sacrifice the audio quality. It makes an ideal candidate for a space-optimized archival pipeline. It can be the source file for creating the Compressed and lossy formats further.

We are interested in exploring the Deep neural network architectures for achieving greater compression than existing methods with minimal loss of audio perceptual quality. Variational Autoencoder and Encoder architectures are appropriate for this objective. They typically have a conical encoder reducing the size of the input data to latent space and then a decoder which acts as a reconstruction block.

In a well-trained VAE/AE the latent space is a compressed representation of the input data. Trained on huge corpora, there are many popular embeddings like Word2Vec, Glove and Universal Sentence Encoder used for a variety of tasks. The latent space is learned in an unsupervised fashion, minimizing reconstruction loss.

Like other popular approaches [2], [3], [5], [6] we plan to adopt the speech data preprocessing using Mel Spectrogram, STFT, and RFFT on the TIMIT data. We plan to train multiple architectures (VAE & AE) using Fully Connected Networks, CNNs and evaluate the performance using MSE.

The existing methods employ autoencoders for compression and require comparably complicated and larger models which lead to higher training and inference time. Also, the existing methods for efficient data compression are computationally more expensive and require huge amounts of computational resources to achieve robust performance. On the contrary, we plan to train a simple model that would work on limited resources and compress the speech data with a low inference time.

2. RELATED WORK

Duan et al. [2] proposed a VAE based multi perceptron network to encrypt images into a lower dimension. The authors tracked PSNR and MSE to evaluate the

reconstructed images. They claim the model to be fast and easy to encrypt and reconstruct with low distortion in the decompressed images. The model is developed on image data and focuses on the encryption abilities of the latent space.

Srihari Kankanahalli [3] worked on an End to End Optimized speech coding using Deep Neural networks. It is a wideband speech coding pipeline consisting (compression, quantization, and decompression). The author evaluates the reconstruction by computing the PESQ [9] score on the decompressed speech data. The pipeline works over the bit rates (9kbps ~ 24 kbps). It is not focused on the compression ratio, rather emphasizes on the reconstruction quality.

Mohit Goyal et al. [6] proposes using RNN and encoder-based pipeline for lossless compression of sequential data. The model works well for sequence data like genome sequences and DNA sequences etc. They use the Compression ratio for different datasets as an evaluation metric. It is not tested on speech or audio data or other sequential data.

Amund Vedal [5] proposed the use of VQ VAE for compression of audio spectrograms using a discrete quantization method. The proposed model achieves a 9x compression with good reconstruction quality. They employed MSE as an evaluation metric for the reconstructed speech samples. The model produces some distorted results for a few samples in the data.

3. METHODOLOGY

We have implemented multiple methods in a quest to achieve a reasonable reconstruction of the speech data. The following sections describe the methodology of the individual approaches.

3.1. DATA

For the training and testing, we have used TIMIT acoustic-phonetic Continuous Speech Corpus. There are 6300 speech samples of 630 speakers with 10 utterances each in the 8 major dialects of English in the USA. The files are stored in a wav format with a sampling rate of 16000. It is developed for evaluating Automatic speech recognition systems.

3.2. PRE-PROCESSING

3.2.1. Audio transformation

We converted the audio files in the '.wav' format to time-domain signals. The time-domain signals were further transformed in Mel spectrograms and STFT spectrograms to be fed in as input to CNN based autoencoders after normalization and scaling. Griffin-Lim [7] algorithm was used to convert the generated output spectrogram (retrieved fter de-normalization and rescaling) to the time-domain signal.

We computed Real Fast Fourier Transform for the speech signals to feed the computed values as input to fully connected neural network. The generated output was converted back to time-domain from frequency domain using Inverse Real Fast Fourier Transform.

3.2.2. Normalization

For Mel spectrogram and STFT, we have normalized the training data using log wrapper.

$$A\_norm = \log(A + c)$$

Where 'A' is spectrogram data and 'c' is a small constant that can be tuned to transform the values in spectrograms to a close to normal distribution.

3.2.2. Scaling

We have scaled the STFT spectrograms to the range [0, 1] using min-max scaling.

The sound signals are scaled to [-1, 1] in the time domain and RFFT is applied.

The effect of normalizing by setting the parameter c and scaling data is as follows.

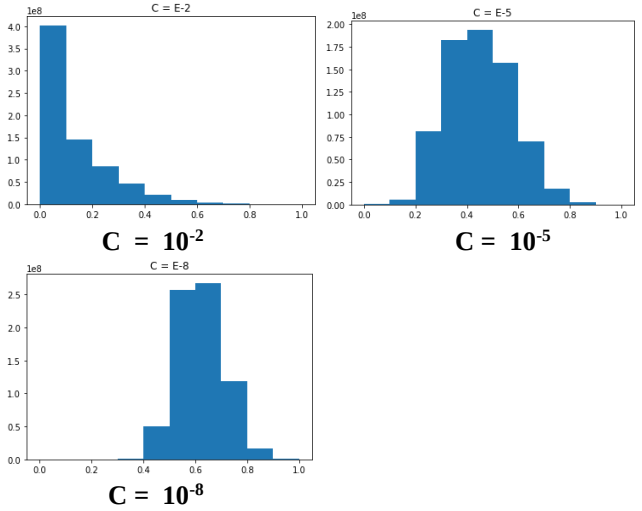

Fig 1: Effect of C on the shape of the distribution for the STFT data of TIMIT Corpus.

By choosing the right c value in the equation the input data can be transformed into a normal distribution. We have selected the c value for the distribution close to the normal

distribution and then scaled the data to [0, 1] using min-max.

3.3. AUTOENCODER STFT

The speech files after being loaded in as time-domain signal are converted into STFT spectrograms with n_fft = 1024 and hop_length = 512. Each spectrogram is then padded (if required) and divided into chunks of fixed dimensions to be given as input to the model. The dimensions for each chunk is (513,29). The data is then normalized and scaled between 0 and 1 before being given as input to the autoencoder. The training data in TIMIT is divided into training and validation sets. The size of the validation set is 100 speech files and the size of the training set is 4520 speech files.

3.3.1 Architecture

The autoencoder model consists of a total of eight layers of convolutional neural networks with the first four layers acting as an encoder and the last four layers acting as a decoder. The details regarding the architecture can be seen in Fig. 2. We use dropout regularization after second, third, fourth, fifth and sixth convolutional neural network layers in the autoencoder with a dropout rate of 25%. All the hidden layers use Leaky RELU as activation function and the output layer uses Sigmoid as the activation function. All the weights in the model are initialized with He initialization for faster convergence. The model loss is reduced ADAM optimization [8]. Mean Squared Error loss is used.

Layer	Filter Size	Stride Dimension	Output Size (Initial Input-513x29x1)
conv2D	5x5	1x1	513x29x64
conv2D	5x5	2x2	257x15x64
conv2D	3x3	2x2	129x8x64
conv2D	3x3	2x2	65x4x10
conv2D_Transpose	3x3	2x2	129x8x64
conv2D_Transpose	3x3	2x2	257x15x64
conv2D_Transpose	5x5	2x2	513x29x64
conv2D_Transpose	5x5	1x1	513x29x1

Fig 2: Autoencoder STFT CNN architecture

3.3.2 loss function

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2$$

The model was trained by minimizing the mean squared loss (MSE) at each update. The loss was computed between the input signal and the reconstructed signal.

And then averaged over the chunks in each speech sequence.

3.3.3 Training and Testing steps

1. The model is trained on the data with a batch size of 100 chunks of size (513,29) for 1000 or more epochs before being able to learn the compression representation and being able to generate satisfactory results.

2. Calculate performance on validation data after every 10 epochs.

3. Once the model is trained, the output received from the test files given as input (in chunks of shape 513x29) to the model after normalization and scaling, is de-normalized and rescaled to its original scale.

4. The denormalized and rescaled (513,29) chunks which are model output are put together and padding (if any) is removed to get the dimensions that are the same as initial input spectrogram retrieved from time domain of respective test file.

5. We use the Griffin-Lim [7] algorithm to approximate the phase and convert the reconstructed spectrogram into a time-domain signal.

6. The time-domain signals are then evaluated using the Signal-to-Noise Ratio (SNR) and Perceptual Evaluation of Speech Quality scores (PESQ) [9]. The example spectrograms plotted in Fig. 5 and Fig 6 can be referred to for a comparison between original and reconstructed spectrograms.

.3.3. AUTOENCODER RFFT

Load the WAV files and scale the time domain signal to [-1, 1]. Apply Real fast Fourier transform to the signal and divide the signal into chunks of size 12348. We appended them and store in an array.

3.3.1 Architecture

A simple feed-forward neural network with fully connected layers is used for training the model. [Fig 3]

Layer	#Hidden Units
Input Layer	12348
Layer 1	8500
Layer 2	3500
Layer 3	2750
Layer 4	3500
Output Layer	12348

Fig 3: Autoencoder RFFT DNN Architecture

The latent layer dimension was selected by checking the performance of the reconstruction. Layer 1, Layer 2 and Layer3 form the encoder part of the network, the rest of the layers form the decoder. In each layer, we used the He Initialization for the weights. Also, we used RELU activation wrapper for the initial layers, the final layer is left out to be linear activation. To restrict the weights from increasing, we employed L2 Regularization in each layer with a lambda = 0.0001. We used ADAM [8] optimizer to update the weights and minimize the loss function.

3.3.2 Loss function

The model was trained by minimizing the mean squared loss (MSE) at each update. The loss was computed between the input signal and the reconstructed signal.

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2$$

And then the loss is averaged over the chunks in the training corpus.

3.3.3 Training and testing steps

1. The model is trained on the data with a batch size of 1 speech sample with chunks of size (12348) for 5000 updates for a decent reconstruction of the time domain speech signal

2. Calculate performance on validation data after every 10 updates. The training curve couldn't be plot for validation for each update since the validation inference is expensive at each update on Google Colab. There is a fluctuation in training curve since the variable length chunks are passed in each batch. [Fig 8]

3. Once the model is trained, the output received from the test files given as input (in chunks of shape 12348) to the model after scaling, is de-normalized and rescaled to its original scale.

4. The denormalized and rescaled (12348) sized chunks which are model outputs are concatenated to get the dimensions that are the same as initial input RFFT retrieved from the time domain of respective test files.

5. We use the IRFFT algorithm to convert the reconstructed signal from the network into a time-domain signal.

6. The time-domain signals are then evaluated using the Signal-to-Noise Ratio (SNR) and Perceptual Evaluation of Speech Quality scores (PESQ) [9]. The example spectrograms plotted in Fig. 9 and Fig 10 can be referred to for a comparison between original and reconstructed time-domain signals.

4. RESULTS

4.1 STFT CNN AE MODEL RESULTS

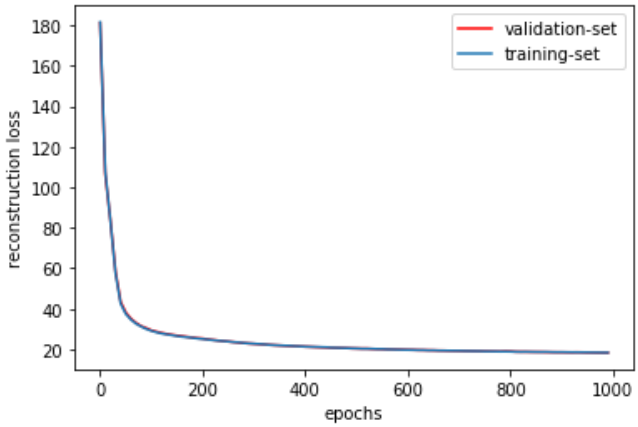

Fig 4: Training curve

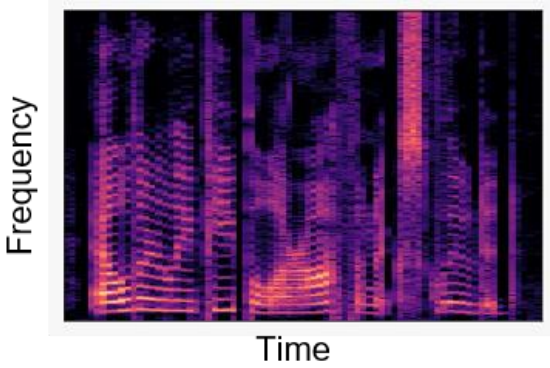

Fig 5: Original STFT

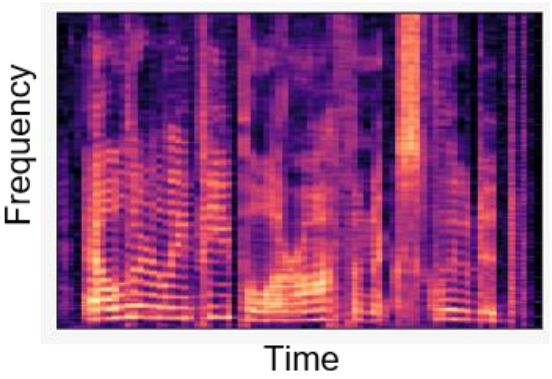

Fig 6: Reconstructed STFT

4.1.1. Evaluation metrics

Metric	Value
Mean SNR	6.555
Mean PESQ Narrow Band	1.751
Mean PESQ Wide Band	1.458

Fig 7: Evaluation metrics for Autoencoder STFT model

4.2 RFFT FCN MODEL RESULTS

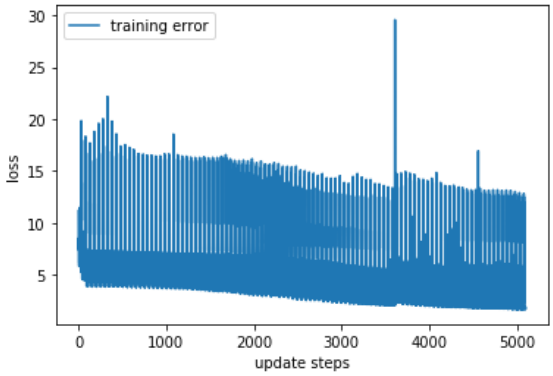

Fig 8: Training curve

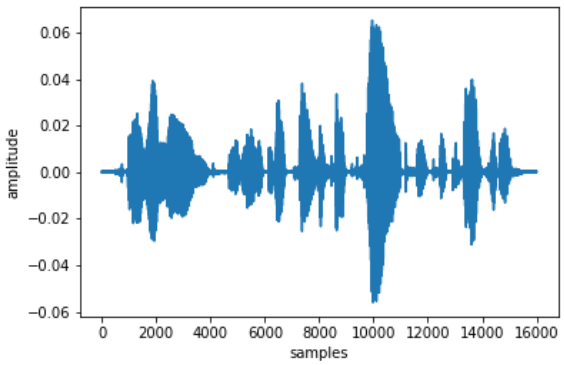

Fig 9: Original time-domain input

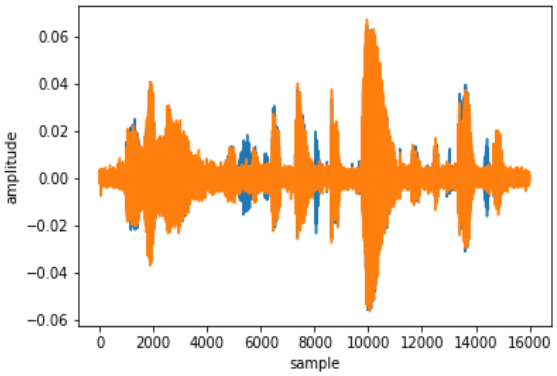

Fig 10: Reconstructed time-domain sample

4.2.1. Evaluation Metrics

Metric	Value
Mean SNR	-2.006
Mean PESQ Narrow Band	1.641
Mean PESQ Wide Band	1.1785

Fig 11: Evaluation metrics for Autoencoder RFFT model

The PESQ [9] score for the presented CNN based autoencoder and fully connected neural network based autoencoder approaches are comparable to each other. Although, the SNR values tell a different story that agrees with the evaluation from listening to the reconstructed speech. STFT approach has no background noise, RFFT adds a little noise to reconstructed speech. The compression ratio for the CNN based autoencoder is 6:1 and for the fully connected neural network based autoencoder is 4:1.

7. DISCUSSION

We initially attempted to train Variational Autoencoders and Beta Variational Autoencoder on Mel spectrograms and STFT spectrograms. Although the latent loss reduced steadily while training the models on the training data, the reconstruction loss would not converge low enough for the Variational Autoencoders to be able to produce output that was relevant and interpretable.

As we tend to penalize the model for learning latent representations that are not likely to be drawn from a standard normal distribution, we restrict the amount of information that can possibly be stored in the latent space. Hence Variational Autoencoders are not practical for the

purpose of compression and don't give comparable performance to that of simple Autoencoders.

From the experiments conducted to find suitable architecture and hyperparameters to get satisfactory results, we have observed that while training autoencoders on spectrograms, the appropriate data normalization, and standardization techniques bring significant performance improvements. Also, STFT spectrograms provide better results as compared to Mel spectrograms with the quality of the results being heavily dependent on the STFT parameters.

One reason behind lacking robust performance while using Mel spectrograms might be that converting Mel spectrograms to time-domain signals leads to a higher loss in information as compared to converting STFT spectrograms to time-domain signals using Griffin-Lim [7] algorithm. This is because STFT spectrograms are approximated from Mel spectrograms before converting the STFT spectrogram to time-domain signal using the Griffin-Lim [7] algorithm (approximates phase) while converting Mel spectrograms to time-domain signals.

Another observation from training deep learning models on speech signals was that there was a high level of difficulty in intuitively interpreting the progress being made by the neural networks while training the neural networks. Unlike training autoencoders on images, where the reconstructions are to an extent self-explanatory compared to reconstructions of spectrograms or RFFT values by autoencoders while training using spectrograms and RFFT values of speech data due to lack of direct interpretability in spectrograms and RFFT values.

Furthermore, from training multiple architectures before finding a suitable architecture we observed that while training the autoencoder on spectrograms with a fully connected final layer in the encoder and a fully connected first layer in decoder lead to lack of convergence of reconstruction loss. Intuitively, this can be attributed to the loss in spatial information while transitioning from a CNN layer to a fully connected neural network layer. Hence, we chose to encode the input data in the CNN layer output. Additionally, we found the use of max-pooling layers in encoder discouraging in terms of reconstruction loss and instead resorted to using a stride of dimension (2,2) with the 'same' padding to reduce the dimensions of the output in the CNN layers of the encoder.

8. FUTURE WORK

A general framework for better intuition on training deep learning models on audio data need to be developed. Metrics pertinent to the better generalization of deep learning models on audio data need to be identified that can help in creating a better intuitive understanding in terms of improving neural network training on audio data or choosing architectures for audio related tasks.

Extensive exploration of model architectures for lightweight deep learning models is required for the task of audio compression with low inference time. A Lightweight MobileNets inspired autoencoder can be built for compression of data that can be used on edge devices.

Quantitative and Qualitative evaluation of autoencoder performance with different compression ratios for speech data using autoencoders built with different architectures.

Implementing a ConvLSTM [10] based autoencoder architecture. ConvLSTM [10] is a model in which all the operations in LSTM are convolutions. ConvLSTM [10] might be useful for speech-related tasks as it processes both temporal and spatial information. ConvLSTM [10] based autoencoders might produce better reconstruction spectrogram as compared to using a purely CNN based architecture or a purely LSTM based architecture. A study of comparison between LSTMs with CNNs, and ConvLSTM [10] along with their advantages and disadvantages on speech-related tasks can also be insightful.

9. CONCLUSION

Both the proposed approaches, CNN based autoencoder trained on STFT spectrograms and fully connected neural network-based autoencoder trained on RFFT values provide satisfactory results in terms of interpretability in the generated audio files. The fully connected neural network-based autoencoder trained on RFFT values generates reconstructions that retain the vocal qualities of the original signal with minimal vocal distortion while having faint background noise. Whereas, the CNN based autoencoder trained on STFT spectrograms generates reconstructions with very little or no background noise while having slightly distorted vocal features as compared to the original signal. These models are much simpler and smaller than many existing autoencoders for speech compression. Although the smaller architectures and simpler solutions presented in this work as compared to existing complicated methods with large architectures have a performance tradeoff, these methods can be useful in cases where priority is given to interpretability with low latency requirements. Further optimization of these models for lower inference time and lower computational resources is seemingly possible through further tuning and evaluating lightweight models with similar architectures. Due to lower inference time of the proposed approaches and lower computational resources required they can also be used in edge devices with low computational resources and low inference time requirements.

10. ACKNOWLEDGMENT

We would like to take this opportunity to thank all the course staff who supported and helped us in completing this project. Firstly, we thank Prof. Donald Williamson and IU for making this amazing course available for us to register. Also, we would like to thank Prof. Donald Williamson for making the course so informative and for allowing us to choose the project of our liking. We are very grateful for giving us $50 Google cloud credit which has been very instrumental in completing the project in the stipulated time frame. Special thanks to TA (Hai Hu) for helping us with the course. Finally, we thank all the fellow students who made the classes informative and insightful.

11. CONTRIBUTION

We started with data collection and literature review, in which both of us are actively involved. Sahaj worked on the data processing for the CNN based model. Siva Charan worked on the data preprocessing for the RFFT model. Sahaj worked on the CNN based autoencoder development and training. Siva Charan worked on RFFT FCN model development and training. The initial experiments involving Variational Autoencoders for speech compression were carried out by both of us.

12. REFERENCES

[1] Garofolo, J. S., Lamel, L. F., Fisher, W. M., Fiscus, J. G., & Pallett, D. S. (1993). DARPA TIMIT acoustic-phonetic continous speech corpus CD-ROM. NIST speech disc 1-1.1. NASA STI/Recon technical report n, 93.

[2] Duan, X., Liu, J., & Zhang, E. (2019). Efficient image encryption and compression based on a VAE generative model. Journal of Real-Time Image Processing, 16(3), 765-773.

[3] Kankanahalli, S. (2018, April). End-to-end optimized speech coding with deep neural networks. In 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) (pp. 2521-2525). IEEE.

[4] Hsu, W. N., Zhang, Y., & Glass, J. (2017). Learning latent representations for speech generation and transformation. arXiv preprint arXiv:1704.04222.

[5] Amund Hansen Vedal (2019) “Unsupervised Audio Spectrogram Compression using Vector Quantized Autoencoders Master thesis”

[6] Mohit Goyal, Kedar Tatwawadi, Shubham Chandak and Idoia Ochoa “DeepZip: Lossless Data Compression using Recurrent Neural Networks”

[7] D. Griffin and J. Lim, “Signal estimation from modified short-time Fourier transform,” IEEE Transactions on Acoustics, Speech, and Signal Processing, vol. 32, no. 2, pp. 236–243, 1984.

[8] Kingma, Diederik P and Ba, Jimmy Lei. Adam: A method for stochastic optimization. arXiv
preprint arXiv:1412.6980, 2014.

[9] A. Rix, J. Beerends, M. Hollier, A. Hekstra, "Perceptual evaluation of speech quality (PESQ)-A new method for speech quality assessment of telephone networks and codecs", Proc. IEEE Int. Conf. Acoust. Speech Signal Process., vol. 2, pp. 749-752, 2001

[10] Xingjian Shi, Zhourong Chen, Hao Wang, Dit-Yan Yeung, Wai-Kin Wong, and Wang-chun Woo.” Convolutional LSTM network: A machine learning approach for precipitation nowcasting”. CoRR,2015.